CONFIDENTIAL. Limited circulation. For review only.

# Spatio-Temporal Avoidance of Predicted Occupancy in Human-Robot Collaboration

Jared Flowers, Marco Faroni, Gloria Wiens, Nicola Pedrocchi

*Abstract*— **This paper addresses human-robot collaboration (HRC) challenges of integrating predictions of human activity to provide a proactive-n-reactive response capability for the robot. Prior works that considered current or predicted human poses as static obstacles are too nearsighted or too conservative in planning, potentially causing delayed robot paths. Time-varying prediction of human poses enables robot paths that avoid anticipated human poses at necessary times. The main contribution herein is a proactive path planning method, denoted STAP, that uses spatio-temporal human occupancy maps to find trajectories that anticipate the human movements, allowing the robot passage without stopping. For example, STAP can anticipate delays from robot speed restrictions required by ISO15066 speed and separation monitoring (SSM). STAP also proposes a sampling-based planning algorithm based on RRT* to solve the spatio-temporal motion planning problem and find paths of minimum expected duration. Experimental results show STAP generates paths of shorter duration and greater average robot-human separation distance throughout tasks. Additionally, the STAP method more accurately estimates robot trajectory durations in HRC, which are useful in arriving at proactive-n-reactive robot sequencing.**

## I. Introduction

A prevalent challenge in human-robot collaboration (HRC) is providing robot(s) proactive-n-reactive capability to correctly anticipate and respond to humans working in close proximity. In some contexts, such as manufacturing, robot(s) and human(s) typically perform cyclic tasks, in which human motion is likely repetitive and predictable. If HRC controller algorithms, including motion planners, can efficiently leverage such knowledge, then the robot can proactively avoid anticipated interruptions. Additionally, the robot could take advantage of predicted windows of time that are free of obstruction to move safely and seamlessly among humans. A motion planner that utilizes the predicted motions of humans can also estimate a 'robotic task completion time' metric useful in selecting the next robot task to perform. Via time parameterization of the waypoint sequence, the planner can further modify the robot motion in a way that reduces the amount of time a robot spends in close proximity to a human throughout each given task, allowing greater human comfort.

State-of-the-art variations of Rapidly Exploring Random Trees (RRT) and Probabilistic Roadmaps have included features that allow reaction to an obstacle's current pose while computing fast enough for online planning [1-5]. These recent variations can only consider current, static poses of dynamic obstacles. Recent human-aware planners consider static, anticipated human volumes and robot speed reduction due to human proximity to generate paths specifically beneficial to HRC [6-8]. Human-aware planners also consider dynamic obstacles as having anticipated, static poses. Since these planners only consider static obstacle poses and do not consider obstacles' time varying motion, they are nearsighted in planning. This near-sightedness allows obstacles to drive the robot toward temporary entrapments and production delay. To show the importance of considering time-varying predictions of human poses, consider a workcell with two assembly stations, a robot that services both stations, and a human that alternates between stations. If the robot path planner considers human pose as static, then the human and robot could interfere with each other as the human moves between stations. The method herein permits robot paths that proactively account for the anticipated human motion between stations, allowing uninterrupted robot and human motion.

This paper proposes a method to reduce the interference between humans and robots during collaborative tasks. The proposed approach includes short-term predictions of the humans' occupancy in a motion planning problem and searches for a trajectory that avoids potential collisions. While previous methods have also used time-based cost functions, the method herein has key differences from previous works. First, this method uses predicted human motion in 3D space to estimate time-varying human occupancy in a robotic workcell. Second, it formulates the trajectory planning as an optimal motion planning problem minimizing the trajectory execution time under the spatio-temporal avoidance constraints. A variation of RRT* is developed to solve such problem. Third, the method incorporates the *speed and separation monitoring (SSM)* rules from the ISO15066 standard to estimate the effect of predicted human poses on robot speed and estimated trajectory duration [9]. Experiments in different manipulation scenarios show that the trajectories planned with the proposed spatio-temporal approach have a significantly shorter execution time and larger separation distance between humans and robots during close collaboration.

The following sections include a discussion of the STAP method's key steps, experimental validation, results, and

*Funding was provided by the NSF/NRI: INT: COLLAB: Manufacturing USA: Intelligent Human-Robot Collaboration for Smart Factory (Award I.D. #:1830383).

J. Flowers and G. Wiens are with Mechanical & Aerospace Engineering, University of Florida, Gainesville, FL 32611 USA (e-mail: jared.flowers@ufl.edu, gwiens@ufl.edu).

M. Faroni is with the Department of Robotics of the University of Michigan, Ann Arbor, MI 48109 USA (e-mail: mfaroni@umich.edu)

N. Pedrocchi is with Istituto di Sistemi e Tecnologie Industriali Intelligenti per il Manifatturiero Avanzato Consiglio Nazionale delle Ricerche, Milan, Italy (e-mail:, Nicola.pedrocchi@stiima.cnr.it).

Manuscript 373 submitted to 2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN). Received March 31, 2023.

conclusions. Section III includes subsections: III.A) detailing the predicted human model, III.B) avoidance intervals, III.C) incorporation of idle times owed to safety stops and slowdowns, III.D) the variation of RRT* to utilize avoidance intervals, and III.E) STAP time-parameterization.

## II. RELATED WORKS

Prior planning methods incorporate predicted obstacle motion to proactively avoid anticipated obstructions. One option is to dynamically modify the robot's trajectory based on the immediate human motion [10,11]. Another method generates collision free waypoints when time is included in the configuration space, and then uses a local planner to ensure reachability [12]. Other works define safe intervals for passage through configuration nodes to avoid obstacles when generating 2D robot paths, in which both obstacles and robot are defined as points [13,14]. One planning framework models probabilistic human motion sequences using time sequences of Gaussian Mixture Models (GMMs) [15,16]. It uses the STOMP path optimizer to deform straight-line paths so robot penetration into the human's volume along the sequence is minimized. Other methods use optimization frameworks to generate robot trajectories that maintain a distance between the robot's trajectory and learned time-sequences of human arm motion [17,18]. Some approaches avoid anticipated points of collision for the entire task by considering previously occupied space or predicted human occupancy volumes (HOVs) as static obstacles throughout the task [6,19]. The method presented, herein, differs from prior work in that it utilizes predicted dynamic motions of humans over an entire task rather than static definitions or instantaneous estimates.

In an HRC manufacturing workcell, observations of human task repetitions facilitate prediction of human motion, providing possible inputs to the method presented in this paper. One such motion prediction method uses GMMs to generate time sequences of probabilistic models for human arm extension trajectories [20,21]. Then it uses the GMMs for classification and extrapolation for current human motion. GMMs were also used to learn a linear dynamic model to represent human trajectories [22,23]. Probabilistic principal component analysis has also been used to learn motion models for human trajectories, detect motion onset, estimate human speed, and select a motion model to infer a future trajectory [24]. Methods such as Risk of Passage and the Swept Volumes method anticipate volumes at the intersection of robot paths and predicted human motion [25,26].

## III. METHOD

The goal of this work is to develop a cost function and sampling-based robot path planner variation that considers anticipated, time-varying motion of humans. STAP first generates a spatio-temporal human occupancy map to encode anticipated human poses and anticipated times of occupancy. Then the STAP method considers the occupancy map as well as estimated speed reductions enforced by an *SSM* safety controller to estimate the time required for a robot to go between pairs of robot configurations. Finally, a variation of RRT* is proposed that creates a graph of nodes and estimated path costs using the STAP cost function. This enables the planner to output the trajectory of minimal duration considering anticipated time-varying human occupancy. Fig. 1 shows how the proactive motion planning provided by STAP fits into an overall robot control scheme as the green blocks. The blue block is the motion prediction method providing input to STAP. Fig. 1 includes the *SSM* safety controller as the orange block which limits robot speed along the planned trajectory based on real-time robot/human separation distance.

### A. Predicted Spatio-Temporal Human-Occupancy Map

The first step of the STAP method is to generate a spatio-temporal occupancy map in cartesian space. Each point of the map will store a set of avoidance intervals, which are the time intervals when any human body part will occupy that point. The input to this method is a time series defining human motion in terms of the pelvis location, quaternions of each human body part (aka limb, human skeletal link) w.r.t. the world-z axis, link lengths, and link radii for the human for each time step. At each time step of the human motion, quaternion forward kinematics is performed using the pelvis location, quaternions, and link lengths to determine the joint locations of the human. Then, the human shape is generalized to a cylinder for each link, shown as green cylinders in fig. 2A. Next, the link cylinders are fit to a discretized 3D grid to generate the blue point cloud in fig. 2B. The cartesian workspace surrounding the robot is denoted by $\mathcal{W}$. Each occupied point in $\mathcal{W}$ is assigned the time of occupancy from the human motion sequence. This results in a set of cartesian points occupied at a list of time steps, denoted $\mathcal{H}$. The set $\mathcal{H}$ can then be used to generate intervals of time over which each point in $\mathcal{W}$ is occupied. Each interval has start time $t_s$ and end time $t_f$. If a point in $\mathcal{H}$ is occupied at consecutive time steps, then the first consecutive time of occupancy is the start time of an interval and the last consecutive time is the end time. A single point in $\mathcal{H}$ can have multiple intervals, denoted $[t_s, t_f]$, over which the point is occupied.

### B. Avoidance Intervals

The points in $\mathcal{W}$ must inherit occupancy intervals from $\mathcal{H}$. Interval start and end times for a point in $\mathcal{W}$ are denoted $t_{s_{o_i}}$

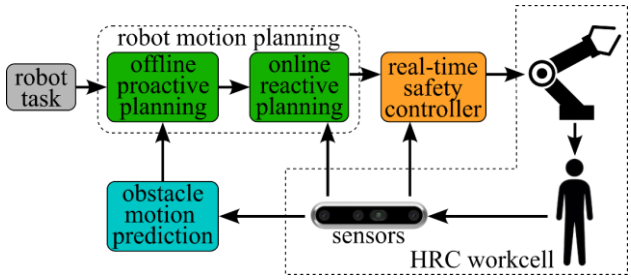
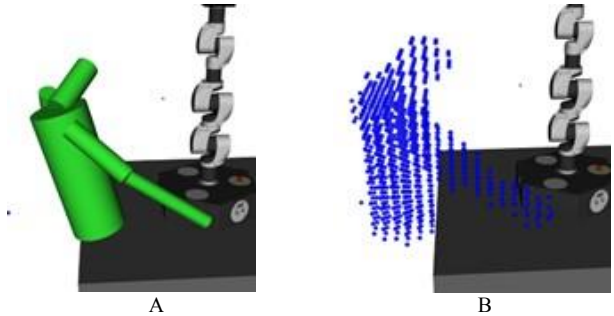



Figure 1. The high-level control loop for an HRC workcell.



A B

Figure 2. (A) Human pose generalized to a set of cylinders. (B) Human cylinders generalized to a point cloud.

and $t_{f_{o_i}}$, respectively, for the $i^{th}$ interval resulting from the $o^{th}$ obstacle at that point. The time intervals of occupancy are:

$$\mathcal{A}_{o_i}(x,y,z) = \begin{cases} \left[t_{s_{o_i}}, t_{f_{o_i}}\right] & t_{f_o} < t_{end_o} \\ \left[t_{s_{o_i}}, \infty\right) & t_{f_o} = t_{end_o} \end{cases}, \quad (1)$$

$$\mathcal{A}(x,y,z) = \cup_o \cup_i \mathcal{A}_{o_i}(x,y,z)\ \forall i \in o, \forall o \in (x,y,z), \quad (2)$$

where $\mathcal{A}_{o_i}(x,y,z)$ is the $i^{th}$ avoidance interval for the $o^{th}$ obstacle occupying point $(x,y,z)$, and $t_{end_o}$ is the final time step of the predicted motion sequence for the $o^{th}$ obstacle. The $\mathcal{A}(x,y,z)$ is the union of all intervals for which humans occupy point $(x,y,z)$. If point $(x,y,z)$ is occupied by the $o^{th}$ obstacle at the end of the motion prediction for that obstacle, then it must be assumed that point $(x,y,z)$ is occupied by the $o^{th}$ obstacle for the remainder of time. This leads to the notion of a last time of passage for some points in cartesian space:

$$t_{lp_{o_i}}(x,y,z) = \begin{cases} \infty & t_{f_o} < t_{end_o} \\ t_{s_{o_i}} & t_{f_o} = t_{end_o} \end{cases}, \quad (3)$$

$$t_{lp}(x,y,z) = \max_{o \in (x,y,z)} \max_{i \in o} t_{lp_{o_i}}(x,y,z), \quad (4)$$

where $t_{lp_{o_i}}(x,y,z)$ is the last time passage is allowed due to the $i^{th}$ interval for obstacle $o$ at cartesian point $(x,y,z)$. The $t_{lp}(x,y,z)$ is the last time of passage for point $(x,y,z)$, which is the maximum of $t_{lp_{o_i}}(x,y,z)$ over all intervals for which humans occupy point $(x,y,z)$.

After determining the spatio-temporal occupancy map, a sampling-based planner can generate a trajectory for the robot. The planner generates a new node, having robot configuration $\boldsymbol{q_c}$. Each robot configuration has dimension equal to the degrees of freedom (DOF) of the robot. Connections between the new node and each of its neighbors, having robot configuration $\boldsymbol{q}_p$, are considered. Robot configurations with spacing $\Delta\boldsymbol{q}$ between $\boldsymbol{q}_p$ and $\boldsymbol{q_c}$ must be checked for collision. Let $\boldsymbol{FK}(\boldsymbol{q_p}, \boldsymbol{q_c})$ represent the poses of the robot determined by forward kinematics at configurations between $\boldsymbol{q_p}$ and $\boldsymbol{q_c}$. If an intersection of points in $\mathcal{W}$ and $\boldsymbol{FK}(\boldsymbol{q_p}, \boldsymbol{q_c})$ exists, then the set of avoidance intervals for the intersecting points will be added to the set of avoidance intervals and last pass time for the connection between $\boldsymbol{q_p}$ and $\boldsymbol{q_c}$ in the planner's node graph ($\mathcal{G}$), denoted $\mathcal{A}(\boldsymbol{q_p}, \boldsymbol{q_c})$and $t_{lp}(\boldsymbol{q_p}, \boldsymbol{q_c})$ respectively:

$$\mathcal{A}(\boldsymbol{q_p}, \boldsymbol{q_c}) = \cup_{(x,y,z) \in \boldsymbol{FK}(\boldsymbol{q_p}, \boldsymbol{q_c})} \mathcal{A}(x,y,z), \quad (5)$$

$$t_{lp}(\boldsymbol{q_p}, \boldsymbol{q_c}) = \max_{(x,y,z) \in \boldsymbol{FK}(\boldsymbol{q_p}, \boldsymbol{q_c})} t_{lp}(x,y,z). \quad (6)$$

Next, if $\mathcal{A}(\boldsymbol{q_p}, \boldsymbol{q_c})$ is not an empty set, then the minimum time to reach node $\boldsymbol{q_c}$ from the start configuration while avoiding potential collision between $\boldsymbol{q_p}$ and $\boldsymbol{q_c}$ must be determined. First, the minimum time required to travel between $\boldsymbol{q_p}$ and $\boldsymbol{q_c}$ is:

$$t(\boldsymbol{q_p}, \boldsymbol{q_c}) = \max_{i \in [1,n]} \left| \frac{\boldsymbol{q_c}[i] - \boldsymbol{q_p}[i]}{\bar{\boldsymbol{q}}[i]} \right|, \quad (7)$$

where $\bar{\boldsymbol{q}}$ is the vector of maximum velocities for each robot DOF and $i$ iterates over each DOF. Each node in $\mathcal{G}$ must store the minimum arrival time for the robot to reach its configuration from the start configuration, denoted $t_{arr_V}$ for node $V$. The $t_{arr}$ for the start node is set to zero. Then the minimum time to reach $\boldsymbol{q_c}$ via $\boldsymbol{q_p}$ is $t_c$:

$$t_c = t_p + t(\boldsymbol{q_p}, \boldsymbol{q_c}), \quad (8)$$

where $t_p$ is initially considered to be $t_{arr}$ for the node at $\boldsymbol{q_p}$. This creates a time interval for passage from $\boldsymbol{q_p}$ to $\boldsymbol{q_c}$ of $[t_p, t_c]$. If $[t_p, t_c]$ intersects $\mathcal{A}(\boldsymbol{q_p}, \boldsymbol{q_c})$, then $t_p$ is updated according to:

$$t_p = \min_i \{\mathcal{A}_i(\boldsymbol{q_p}, \boldsymbol{q_c})_{end} + t_{pad} : [t_p, t_c] \cap \mathcal{A}(\boldsymbol{q_p}, \boldsymbol{q_c}) \equiv 0\} \quad (9)$$

The $t_{pad}$ is a user-defined time padding constant. The larger $t_{pad}$ is, the more tolerant the plan is to deviations in actual human motion compared to the prediction. If $t_p$ or $t_c$ is greater than $t_{lp}(\boldsymbol{q_p}, \boldsymbol{q_c})$, then the connection is blocked indefinitely and is rejected. An example when this could occur is if the human doesn't back away from the robot after a task, blocking the robot's path indefinitely. Every time $t_p$ is updated, $t_c$ must be updated using (8). The $\mathcal{A}(\boldsymbol{q_p}, \boldsymbol{q_c})$ is sorted by increasing interval start time, so the resulting $[t_p, t_c]$ is the earliest time interval when the robot can travel from $\boldsymbol{q_p}$ to $\boldsymbol{q_c}$ without colliding with a predicted human pose. The time to reach $\boldsymbol{q_c}$, denoted $t_c$, can now be considered as the cost to reach $\boldsymbol{q_c}$ for use in finding the optimal path. Now the $t_{arr_{V(\boldsymbol{q_c})}}$ is also updated to be $t_c$ if $t_c$ is less than $t_{arr_{V(\boldsymbol{q_c})}}$.

### *C. Safety-Aware Cost-Function*

The time-based avoidance cost function in STAP could work with any robot, including vehicles or manipulators, and any predicted obstacle motion. When STAP is applied to HRC, the ISO15066 standard defines modes of robot operation to ensure human safety [9]. The *SSM* rules of ISO15066 for HRC requires that a safety controller, shown as the orange block in fig. 1, enforce a robot speed limit proportional to robot/human separation distance. The safety controller stops the robot if separation distance becomes too small. To mitigate potential SSM safety controller slowdown effects, STAP adapts its cost function to anticipate speed reductions based on *SSM* rules applied with the predicted human motion. STAP estimates the effect of *SSM* by calculating the time between nodes using the formulation from [27]. First, the distance between the $i^{th}$ point on the human at time $t_h$ in the predicted motion ($\boldsymbol{h_i}(t_h)$) and the $j^{th}$ robot point at $\boldsymbol{q}$ (denoted $\boldsymbol{FK_j}(\boldsymbol{q})$) is:

$$D_{ij} = \left\| \boldsymbol{h_i}(t_h) - \boldsymbol{FK_j}(\boldsymbol{q}) \right\|. \quad (10)$$

The points of the human in $\boldsymbol{h}(t_h)$ are located at the human's joints and centroids of limbs. Then the maximum speed allowed by the $j^{th}$ robot point relative to the $i^{th}$ human point at time $t_h$ and robot configuration $\boldsymbol{q}$ according to *SSM* is:

$$v_{max}(\boldsymbol{q}, i, j, t_h) = \begin{cases} -a_s T_r - v_h + \sqrt{v_h^2 + (a_s T_r)^2 - 2a_s D_{ij}} & : D_{ij} > \underline{D} \\ 0 & else \end{cases}. \quad (11)$$

The $v_h$ is the human's velocity relative to the robot, $a_s$ is the maximum cartesian deceleration of the robot relative to the human, $T_r$ is the reaction time of the robot, and $\underline{D}$ is a

minimum distance between human and robot allowed for robot motion. The magnitude of the robot's tangential speed in the direction of the human is:

$$v_{robot}(\boldsymbol{q}, i, j, t_h) = \boldsymbol{J_j}(\boldsymbol{q}) \frac{(\boldsymbol{q_c}-\boldsymbol{q_p})}{t(\boldsymbol{q_p},\boldsymbol{q_c})} \cdot \frac{\left(\boldsymbol{h_i}(t_h)-FK_j(\boldsymbol{q})\right)}{\|\boldsymbol{h_i}(t_h)-FK_j(\boldsymbol{q})\|}, \quad (12)$$

where $\boldsymbol{J_j}(\boldsymbol{q})$ is the robot's translational Jacobian for point $j$ on the robot at configuration $\boldsymbol{q}$, calculated by $\boldsymbol{J_j}(\boldsymbol{q}) = \frac{d\boldsymbol{FK_j}(\boldsymbol{q})}{d\boldsymbol{q}}$. The $t(\boldsymbol{q_p}, \boldsymbol{q_c})$ is the nominal duration from (7). Then (7) can be updated with the speed bound ($v_{max}$) allowed by *SSM*:

$$t(\boldsymbol{q_p}, \boldsymbol{q_c}) = \sum_{\boldsymbol{q}=\boldsymbol{q_p}}^{\boldsymbol{q_c}} \max_{i,j} \frac{v_{robot}(\boldsymbol{q},i,j,t_n)}{v_{max}(\boldsymbol{q},i,j,t_n)} \frac{\|\boldsymbol{dq}\|}{\|\boldsymbol{q_c}-\boldsymbol{q_p}\|} t(\boldsymbol{q_p}, \boldsymbol{q_c}), \quad (13)$$

where the range between $\boldsymbol{q_p}$ and $\boldsymbol{q_c}$ is divided into configurations of spacing $\boldsymbol{dq}$ and the time to complete each $\boldsymbol{dq}$ is summed. The $t_n$ is the nominal time to reach $\boldsymbol{q}$:

$$t_n = t_p + \frac{\|\boldsymbol{q}-\boldsymbol{q_p}\|}{\|\boldsymbol{q_c}-\boldsymbol{q_p}\|} t(\boldsymbol{q_p}, \boldsymbol{q_c}). \quad (14)$$

If $v_{robot}/v_{max}$ becomes greater than a user selected threshold, then predicted human poses over a short window into the future ($\Delta t$) can be considered by (13) to see if the delay due to human proximity is predicted to be temporary:

$$\frac{v_{robot}(\boldsymbol{q},i,j,t_n)}{v_{max}(\boldsymbol{q},i,j,t_n)} = \min_{t_s \in [t_n, t_n+\Delta t]} \frac{v_{robot}(\boldsymbol{q},i,j,t_s)}{v_{max}(\boldsymbol{q},i,j,t_s)}. \quad (15)$$

The updated $v_{robot}/v_{max}$ from (15) can be used for $v_{robot}/v_{max}$ in (13) for the configuration in question. This allows STAP to estimate delays induced by humans and the effect of *SSM*. A safety controller external to STAP must enforce real-time compliance with the *SSM* rules.

### *D. Spatio-Temporal Path Planning*

The STAP method includes a variation of RRT* to use the human-avoidance model and time-based cost function. The RRT* planner was a starting point for the proposed variation because it uniformly randomly selects new nodes ($V_{new}$) from within the robot's configuration space and converges to the optimal path. The STAP method can be used with high DOF robots, so random selection of $V_{new}$ ensures exploration of robot configurations. The planner variation in STAP has a few significant deviations from standard RRT*. This variation stores data, such as avoidance intervals, for suboptimal connections for future consideration to reduce computation. Standard RRT* discards suboptimal parent connections.

Standard RRT* includes a mechanism to connect (rewire) from $V_{new}$ to a node near $V_{new}$ ($V_{near}$) if the new connection has less cost than the current connection to $V_{near}$ [28]. This ensures trajectories converge to the optimal trajectory as nodes and connections are added to $\mathcal{G}$. In STAP, $V_{near}$ is rewired to $V_{new}$ if the new connection results in a smaller $t_{arr}$ for $V_{new}$. When this occurs, STAP also considers if connections for which $V_{near}$ is the parent can be improved, up to some number of levels (denoted $N_c$) down the solution tree in the direction of children. Additionally, when a new node is added to the solution tree, all $V_{near}$ in the neighborhood of $V_{new}$ are also checked to see if there are any better parent nodes for $V_{near}$ in the neighborhood. This is necessary for STAP because a rewiring event could reduce the $t_{arr}$ for a node, possibly making it a better parent for another node. This could result in a long chain of parent upgrades and too much computation. By upgrading parent connections only for nodes near $V_{new}$, STAP ensures that if configurations for $V_{new}$ are uniformly randomly selected from the configuration space, then each existing node has equal probability of upgrading its connection to a better parent that would reduce its arrival time. Once a connection to the goal node is found, the optimal sequence of connections from the start node to the goal node, denoted $\sigma^*$, is found to be the sequence that minimizes the goal node arrival time.

### *E. Time Parameterization with Avoidance Intervals*

Once STAP determines the sequence of connections $\sigma^*$, then a time parameterization can also be determined using the connection parent times computed in (8) and (9). The time to reach the goal node would be $t_c$ for the connection whose child is the goal node. Then, looking backwards through $\sigma^*$ starting from the connection to the goal node, the time the robot should reach each connection's parent node ($t_p$) has been computed in (9). The robot velocity between consecutive connections for the $i^{th}$ robot configuration variable was limited according to:

$$|\dot{\boldsymbol{q}}[\boldsymbol{i}]| \leq \left| \frac{\boldsymbol{q_c}[\boldsymbol{i}]-\boldsymbol{q_p}[\boldsymbol{i}]}{t_c-t_p} \right| \ \forall i \in [1, n]. \quad (16)$$

The connection velocity limit reduces the robot's velocity if the next connection is delayed to avoid an anticipated human pose. This prevents the robot from stopping close to a human while waiting for an avoidance interval to pass. The STAP time parameterization also allows $t_p$ for the connection whose parent node is the start node to be greater than zero. This occurs when the robot should wait at the start node before beginning motion to avoid a human within the first connection.

Fig. 3 depicts a path solution found using the above method. Time is shown on the horizontal axis with motion start time at the left and configurations shown on the vertical axis with initial configuration at the top and goal at the bottom. Red (shaded) blocks indicate avoidance intervals where connections between configurations are obstructed. The solid black line and green circles indicate the trajectory through the time/configuration space and waypoints. It shows the connection from $\boldsymbol{q_1}$ to $\boldsymbol{q_2}$ has reduced velocity to prevent stopping at $\boldsymbol{q_2}$ until time $t_2$. The figure also depicts the last pass time ($t_{lp}$) for the $\boldsymbol{q_1} \rightarrow \boldsymbol{q_2}$ and $\boldsymbol{q_2} \rightarrow \boldsymbol{q_3}$ connections.

## IV. EXPERIMENTS

Experiments with simulated and real workcells, each having a robot manipulator and a human worker, were used to validate the STAP method presented in section III. One workcell used for testing included a Comau e.Do 6DOF-6R robot sitting on a table, shown in fig. 4A. Live tests were also performed with this cell equipped with two depth cameras, outlined by blue ovals in fig. 4A, to sense the human's real-time location. Fig. 4A shows the robot can normally rotate a

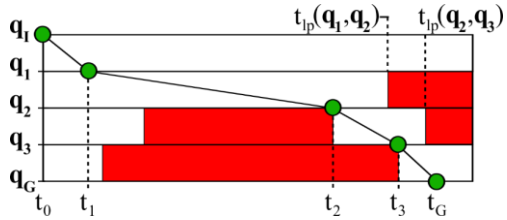
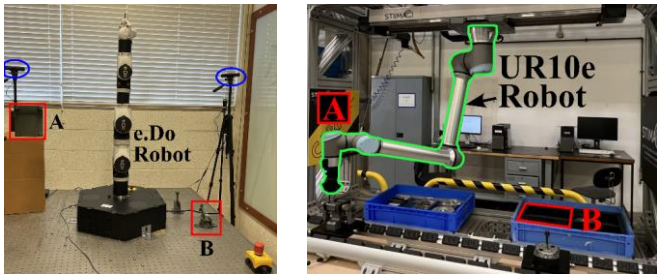



Figure 3. A sequence of nodes/connections from an initial configuration to a goal configuration indicating avoidance intervals for connections.

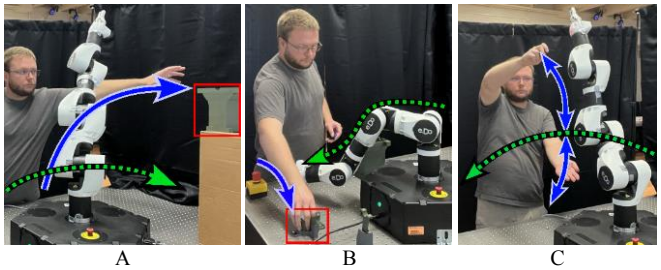
full revolution about its based, but the robot is only allowed to move in the half of the cell nearest the human to ensure robot/human interaction. The second workcell was equipped with a UR10e 6DOF-6R robot mounted hanging over a table, as shown in fig. 4B. These two cells were used because the UR10e has a maximum joint velocity about 4 times greater than the e.Do for any joint, allowing the effect of robot speed on STAP planning to be tested. The two cell configurations also require each robot to take significantly different paths. A computer with an Intel i9 processor used up to 16 CPU cores for path planning.

In tests, the human reached for points in the workspace while the human's motion was recorded as a time series of joint locations. The robot and human performed three tasks together, denoted task A, B, and C. The human reaching targets are shown as red rectangles, human wrist motion as solid blue arrows, and robot end-effector motion as dashed green arrows, in fig. 4 and fig. 5. In task A, the human retrieved an item from a shelf, as shown in fig. 5A, while the robot tried to move an item from the right side of the workcell to the left side, from the human's point of view. In task B, the human retrieved an item from the right side of the table and brought it back to table in front of them while the robot tried to move an item from the left side of the table to the right side. The goal pose of the robot's gripper is just below the trajectory of the human's reaching motion, as shown in fig. 5B. Task C generates a narrow passageway of time-varying size through the configuration space by having the human extend the arms in front of them, pointed at the robot, with palms together and then extend the right arm up and left arm down 45°, as shown in fig. 5C, over 5 seconds, then back together over 5 seconds, repeating twice. Meanwhile, the robot moved an item from the left side of the table to the right side. Tasks A and B represent more realistic motions in an HRC industrial workcell, while C presents greater challenge. In all tasks, the robot must pass through the volume the human will occupy during the task. The human motion prevents the robot from reaching the goal without delay. Each task was performed 10 times in the simulated and live e.Do workcells and simulated UR10e workcell. For avoidance of the real-time human, the live tests used real-time human motions sensed with the depth cameras using the method in [29] while the simulations only used a recording of human motion to emulate a real human. The simulations provided results under ideal conditions, not including any other variations such as those from human's real-time motions. The live tests show how the planning methods can accommodate variation in real human motions.




A B

Figure 4. (A) shows the e.Do workcell and (B) shows the UR10e cell.



A B C

Figure 5. Experimental robot/human tasks.

The STAP method was tested with waypoint timing assigned either as: 1) the output times directly from STAP time parameterization in (8) and (9) (denoted "STAP-PT" in results), or 2) the timing generated by applying Iterative Parabolic Time Parameterization (IPTP) to the STAP generated waypoints (denoted "STAP-IPTP" in results) [30]. The IPTP assigns timing so the robot will reach each waypoint as quickly as possible. These two methods only used the "offline proactive planning" block and not the "online reactive planning" block in fig. 1. For a baseline, tests were also performed using the robot trajectory generated by 3 existing planners. The STOMP planner was used like in [15] to deform the direct trajectory, attempting to avoid the predicted human motion with repulsion inversely proportional to robot-human separation distance. The Bi-directional Transition-based Rapidly-Exploring Random Trees (BiTRRT) planner in OMPL repeated planning as the robot approached the real-time human location, making the robot react to the human's real-time pose [31]. The third planner (denoted "HOVs" in results) is a variant of RRT that tried to avoid all 3D points the human will occupy during the task [6]. The planning time allowed for each planner was 60 seconds to allow convergence near to their unique optimal trajectories.

In all tests, a real-time *SSM* safety controller applied robot speed reduction according to (11)-(13). The *SSM* safety controller reduced robot speed to a fraction of planned speed:

$$\dot{\boldsymbol{q}} = \min_{i,j} \frac{v_{max}(\boldsymbol{q},i,j)}{v_{robot}(\boldsymbol{q},i,j)} \frac{\boldsymbol{q_c}-\boldsymbol{q_p}}{t_c-t_p}, \tag{17}$$

where $\boldsymbol{q}$ is the robot's current configuration, $\boldsymbol{q_p}$ and $\boldsymbol{q_c}$ are the configurations at the start and end of a planned connection, respectively, and $v_{max}(q,i,j)$ and $v_{robot}(q,i,j)$ use the humans current pose instead of prediction in (11)-(13). The $\dot{\boldsymbol{q}}$ is the robot velocity allowed by the *SSM* safety mode. The ratio $v_{max}/v_{robot}$ in (17) reduces speed while $v_{robot}/v_{max}$ in (13) increases time duration. The *SSM* parameters were $T_r$, $a_s$, and $\underline{D}$ of 0.15s, 0.1m/s$^2$, and 0.2m, respectively.

## V. Results and Discussion

The results of experiments with the e.Do and the UR10e are shown in fig. 6. Fig. 6A shows the average duration the robot took to perform each trajectory. These results show that using the STAP method with IPTP resulted in trajectories that took 47% less time on average compared to the other methods, with the exception of task A with the e.Do simulation. The average time reduction by using STAP method versus others was 44% for simulated e.Do, 45% for live e.Do, and 52% for simulated UR10e tests. The faster UR10e yields better time reduction results compared to the slower e.Do robot. Fig. 6B shows average robot-human separation distance during tests. Since prior works consider robot/human separation distance an important metric, it is assumed that the average separation distance metric is directly related to the human's level of

comfort with robot motion [8,9-17,27]. The results show that the STAP method with either timing variation ("STAP-IPTP" or "STAP-PT") resulted in higher average robot-human separation distance. This result means the robot spent less time near the human in all tests except Task C in the UR10e Sim.

The estimated trajectory durations output from the STAP method were also closer to the actual durations observed in all tests. The trajectory generated by the STAP method with IPTP took only 13% more time than the estimate, on average. The STAP method using the time parameterization of (8) and (9) had 73% average duration estimation error. The STOMP and BiTRRT planners generated estimation errors of 646% and 876%. Their estimation errors are large because they planned to do the trajectory as fast as possible but had to react to the human. The HOVs planner estimated infinite path time because its robot paths could not avoid all anticipated human occupancy volumes. These results show the STAP method more accurately estimates robot trajectory durations. The duration estimates can be used to schedule robot tasks to better match the sequence of other machines or humans in an HRC workcell. Trajectory duration estimates are also useful in robot task planning where the robot can select a next action that minimizes anticipated delay caused by humans.

Planner performance was also collected for the STAP method on Task C, allotting 500 planner iterations to generate

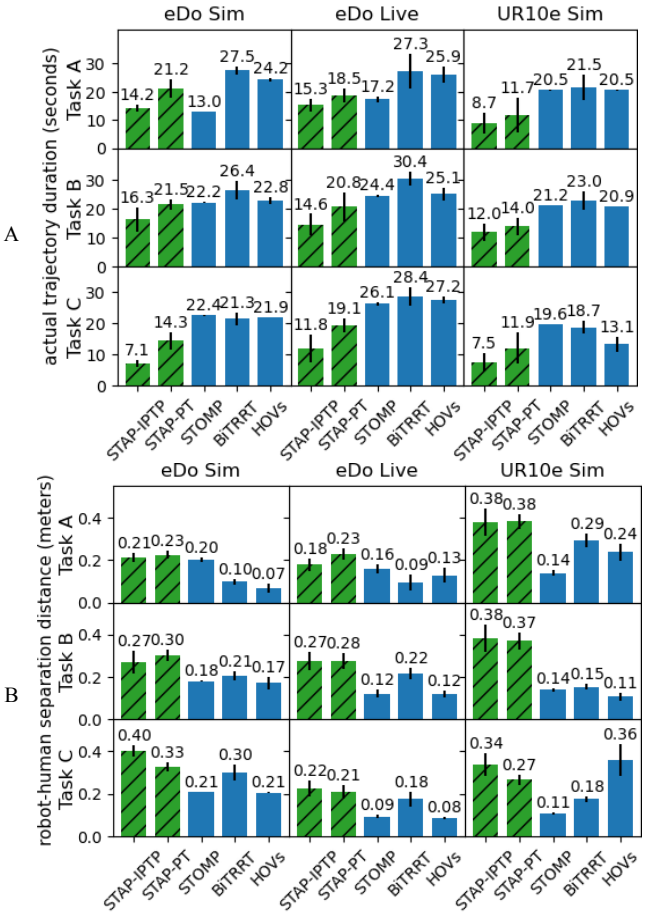



Figure 6. (A) Average trajectory durations and (B) average robot/human separation distance, with mean values indicated above the bars and standard deviations indicated by magnitude of the smaller black bars. Green hatched bars correspond to the STAP method with either timing option.

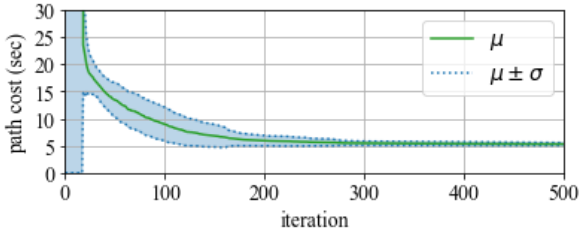



Figure 7. Evolution of average planner cost (trajectory duration estimate) and standard deviation over 500 planning iterations.

the trajectory. Fig. 7 shows evolution of path cost, or estimated trajectory duration, averaged over generation of 100 plans. It indicates the path after 180 iterations was within 15% of optimal. The STAP method performs $N_c n^2$ times more computations per iteration than RRT* due to the STAP planner variation, where $n$ is proportional to the number of nodes at a given iteration and $N_c$ was defined in section III.D. Additional computation stems from processing a predicted human model that is more complex than prior works. The STAP method can also be applied to a manipulator with relatively high DOF with geometrically rich whole arm definition compared to a simplified robot modeled as a point as seen in some prior works. The STAP method improves HRC by providing outputs that mitigate production delay and human discomfort. The trajectory generated by the STAP method provides a less-interrupted robot path, a path that stays farther from the human, and an estimation of the time required for the robot path considering the predicted human motion and slowdowns due to *SSM*. The robotic system could plan motion for a future task ahead of starting it based on the task definition. During online execution of a task, if the real-time human motion becomes too different than the prediction, then the robot could use online re-planning methods to react.

## VI. Conclusion

The presented STAP path planning method combines predicted human motion sequences and proactive robot path planning. It includes a spatio-temporal occupancy map to represent anticipated human poses and a time-avoidance cost function and variation on RRT*. The goal of the STAP method is to mitigate production delays and reduce human discomfort in an HRC workcell. Results showed the STAP method generates trajectories of shorter duration in an HRC setting. Additionally, generated trajectories result in the robot spending less time close to the human. The STAP method also outputs an estimate of the robot trajectory duration, which is useful in arriving at proactive-n-reactive robot sequencing. In future work, more compact forms of the spatio-temporal human occupancy map will be explored to address the computational challenges of STAP.

## Acknowledgment

This work was made possible with funding from the National Science Foundation and collaboration with researchers at the Istituto di Sistemi e Tecnologie Industriali Intelligenti per il Manifatturiero Avanzato, Consiglio Nazionale delle Ricerche (STIIMA-CNR Researcher Director: Irene Fassi). Any opinions, findings and conclusions or recommendations expressed are those of the researchers and do not necessarily reflect the views of the National Science Foundation.

## REFERENCES


[1] L. Yang, J. Qi, D. Song, J. Xiao, J. Han, and Y. Xia. "Survey of Robot 3D path planning algorithms," *Journal of Control Science and Engineering*, pp. 1–22, 2016. DOI:10.1155/2016/7426913.

[2] C. Yuan, G. Liu, W. Zhang, and X. Pan. "An efficient RRT cache method in dynamic environments for path planning," *Robotics Auton. Syst.*, vol. 131, no. 103595, 2020. DOI:10.1016/j.robot.2020.

[3] P.D. Holmes, S. Kousik, B. Zhang, D. Raz, C. Barbalata, M. Johnson-Roberson, and R. Vasudevan. "Reachable sets for safe, real-time manipulator trajectory design," *Robotics: Science and Systems XVI*, vol. 16, pp. 100-113, 2020. DOI:10.15607/RSS.2020.XVI.100.

[4] W. Xinyu, L. Xiaojuan, G. Yong, S. Jiadong and W. Rui, "Bidirectional potential guided RRT* for motion planning," *IEEE Access*, vol. 7, pp. 95046-95057, 2019. DOI:10.1109/ACCESS.2019.2928846.

[5] C. Tonola, M. Faroni, M. Beschi and N. Pedrocchi, "Anytime Informed Multi-Path Replanning Strategy for Complex Environments," *IEEE Access*, vol. 11, pp. 4105-4116, 2023, DOI: 10.1109/ACCESS.2023.3235652.

[6] S. Pellegrinelli, F.L. Moro, N. Pedrocchi, L.M. Tosatti, and T.A.M. Tolio. "A probabilistic approach to workspace sharing for human–robot cooperation in assembly tasks," in *Cirp Annals-Manufacturing Technology*, vol. 65, no. 1, pp. 57-60, 2016. DOI:10.1016/J.CIRP.2016.04.035.

[7] J. Mainprice, E. Akin Sisbot, L. Jaillet, J. Cortés, R. Alami and T. Siméon, "Planning human-aware motions using a sampling-based costmap planner," 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 2011, pp. 5012-5017, doi: 10.1109/ICRA.2011.5980048.

[8] M. Faroni, M. Beschi and N. Pedrocchi, "Safety-Aware Time-Optimal Motion Planning With Uncertain Human State Estimation," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12219-12226, Oct. 2022, DOI: 10.1109/LRA.2022.3211493.

[9] "ISO/TS 15066:2016 Robots and robotic devices – Collaborative Robots," International Organization for Standardization, Geneva, CH, Standard, 2016.

[10] A. M. Zanchettin, N. M. Ceriani, P. Rocco, H. Ding and B. Matthias, "Safety in human-robot collaborative manufacturing environments: Metrics and control," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 882-893, April 2016, DOI: 10.1109/TASE.2015.2412256.

[11] C. Liu and M. Tomizuka, "Algorithmic safety measures for intelligent industrial co-robots," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2016, pp. 3095-3102, DOI: 10.1109/ICRA.2016.7487476.

[12] Y.K. Hwang, P.C. Chen, C. Lee, and M.S. Kim. "Complete motion planner for time-varying environments," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, 1996, pp. 513-519. DOI:10.1109/IROS.1996.570844.

[13] M. Phillips and L. Maxim. "SIPP: Safe interval path planning for dynamic environments," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2011, pp. 5628-5635. DOI:10.1109/ICRA.2011.5980306.

[14] V. Narayanan, M. Phillips and L. Maxim. "Anytime safe interval path planning for dynamic environments," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 4708-4715. DOI:10.1109/IROS.2012.6386191.

[15] J. Mainprice and D. Berenson. "Human-robot collaborative manipulation planning using early prediction of human motion," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 299-306. DOI:10.1109/IROS.2013.6696368.

[16] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2011, pp. 4569-4574, DOI: 10.1109/ICRA.2011.5980280.

[17] Y. Wang, Y. Sheng, J. Wang and W. Zhang, "Optimal collision-free robot trajectory generation based on time series prediction of human motion," in *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 226-233, Jan. 2018, DOI: 10.1109/LRA.2017.2737486.

[18] A. Kanazawa, J. Kinugawa and K. Kosuge, "Adaptive motion planning for a collaborative robot based on prediction uncertainty to enhance human safety and work efficiency," in *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 817-832, Aug. 2019, DOI: 10.1109/TRO.2019.2911800.

[19] R. Hayne, R. Luo and D. Berenson, "Considering avoidance and consistency in motion planning for human-robot manipulation in a shared workspace," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2016, pp. 3948-3954, DOI:10.1109/ICRA.2016.7487584.

[20] J. Mainprice, and D. Berenson. "Human-robot collaborative manipulation planning using early prediction of human motion," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*: 2013, pp. 299–306. DOI:10.1109/IROS.2013.6696368.

[21] R. Luo, R. Hayne, and D. Berenson. "Unsupervised early prediction of human reaching for human–robot collaboration in shared workspaces," *Autonomous Robots*, vol. 42, pp. 631-648, 2018. DOI:10.1007/s10514-017-9655-8.

[22] G.J. Maeda, M. Ewerton, R. Lioutikov, H.B. Amor, J. Peters, and G. Neumann. "Learning interaction for collaborative tasks with probabilistic movement primitives," in *Proc. of the IEEE-RAS International Conference on Humanoid Robots*: 2014, pp. 527-534. DOI:10.1109/HUMANOIDS.2014.7041413.

[23] G.J. Maeda, G. Neumann, M. Ewerton, R. Lioutikov, O. Kroemer, and J. Peters. "Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks," *Autonomous Robots*, vol. 41, pp. 593-612, 2017. DOI:10.1007/s10514-016-9556-2.

[24] T. Callens, T. Van der Have, S. Van Rossom, J. De Schutter, and E. Aertbeliën. "A framework for recognition and prediction of human motions in human-robot collaboration using probabilistic motion models," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5151-5158, 2020. pp. 5151-5158. DOI 10.1109/LRA.2020.3005892.

[25] J.T. Flowers and G.J. Wiens. "Collaborative robot risk of passage among dynamic obstacles," *in Proc. of the ASME International Manufacturing Science and Engineering Conference*, MSEC2021-1977, 2021, 10pp. DOI: 10.1115/MSEC2021-63670

[26] G. Streitmatter, J. Flowers, and G. Wiens, "High fidelity human modeling via integration of skeleton tracking for predictive HRC collision detection," in *Proc. of the ASME International Mechanical Engineering Congress and Exposition*, IMECE2021-68054, 2021, 10pp. DOI:10.1115/IMECE2021-68054.

[27] M. Faroni, M. Beschi, and N. Pedrocchi, "An MPC framework for online motion planning in human-robot collaborative tasks," in *Proc. of the IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA),* 2019, pp. 1555-1558, doi: 10.1109/ETFA.2019.8869047.

[28] S. Karaman and E. Frazzoli. "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846-894, June 2011. DOI: 10.1177/0278364911406761.

[29] J. Flowers and G. Wiens, "Comparison of human skeleton trackers paired with a novel skeleton fusion algorithm," in *Proc. of the ASME Manufacturing Science and Engineering Conference (MSEC),* 2022, pp. 10.

[30] D. Coleman, I.A. Şucan, S. Chitta, N. Correll, "Reducing the barrier to entry of complex robotic software: a MoveIt! case study," *Journal of Software Engineering for Robotics*, vol. 5, no. 1, pp. 3–16, 2014. DOI: 10.6092/JOSER_2014_05_01_p3.

[31] D. Devaurs, T. Siméon, J. Cortés, "Enhancing the transition-based RRT to deal with complex cost spaces," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2013, pp. 4120-4125. DOI: 10.1109/ICRA.2013.6631158.